Überblick

Mischbetrieb: periodisch ↔ aperiodisch bzw. sporadisch

Prioritätsorientierte Einplanung nichtperiodischer Arbeitsaufträge

Übernahmeprüfung (S. 5- 28) sporadischer Arbeitsaufträge

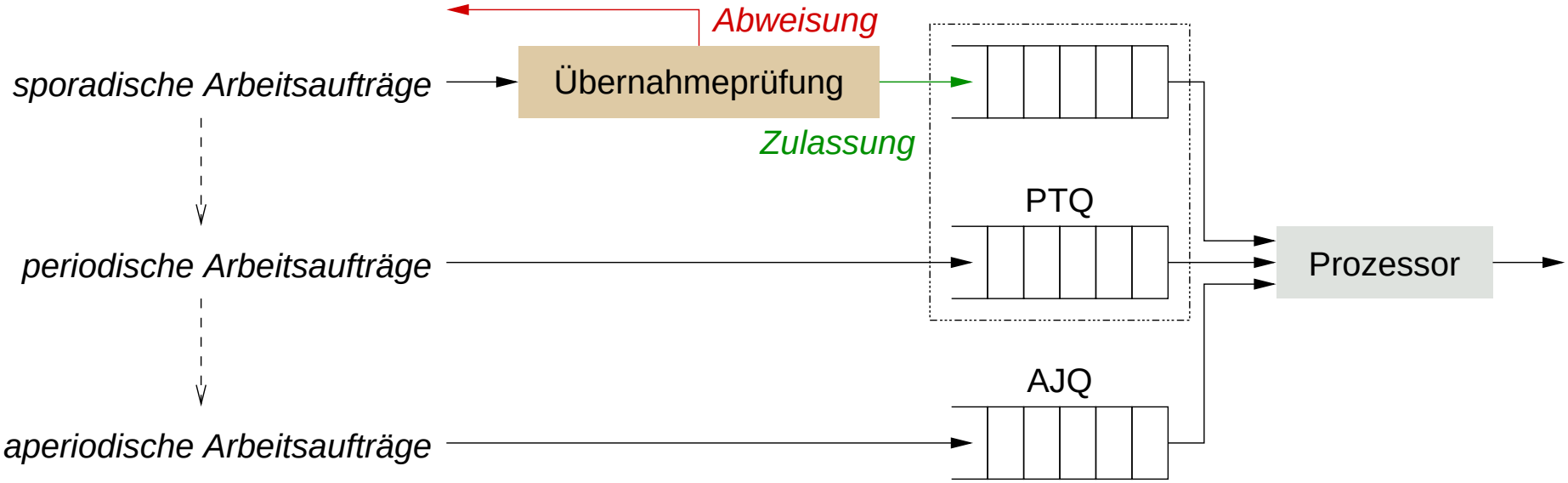
- Entscheidung über Zulassung oder Abweisung des Arbeitsauftrags:
  1. Einplanung eines zugelassenen sporadischen Arbeitsauftrags, so dass sein Abschluss zum vorgegebenen Termin sichergestellt ist
  2. Zusicherung der Termineinhaltung aller periodischen Aufgaben und anderen zuvor bereits zugelassenen sporadischen Arbeitsaufträge
- Parameter: Ausführungszeit und Termin des Arbeitsauftrags

Anwortzeitminimerung aperiodischer Arbeitsaufträge

- bei Termineinhaltung aller periodischen Aufgaben sowie der bereits zugelassenen (eingeplanten) sporadischen Arbeitsaufträge

Prioritätswarteschlangen im Betriebssystem

Variante von MLQ (*multi-level queue*, S. 6- 16)



- bereitgestellte periodische Arbeitsaufträge $\rightsquigarrow$ *Periodic Task Queue*
- sporadische Arbeitsaufträge durchlaufen ein/zwei Warteschlangen:
  1. ausgelöste Arbeitsaufträge müssen ggf. auf Übernahmeprüfung warten
  2. zugelassene Arbeitsaufträge kommen in eine eigene oder die PTQ
- bereitgestellte aperiodische Arbeitsaufträge $\rightsquigarrow$ *Aperiodic Job Queue*

Korrektheit und Optimalität

Einplanungsverfahren terminabhängiger Arbeitsaufträge

korrekter Ablaufplan $\mapsto$ periodische/sporadische Arbeitsaufträge

- analytischer Aspekt: periodische sowie zugelassene sporadische Arbeitsaufträge verpassen niemals ihre Termine
- konstruktiver Aspekt: die Aktionen des Gesamtsystems bewirken für diese Arbeitsaufträge keine Terminverletzungen

optimaler Ablaufplan $\mapsto$ aperiodische/sporadische Arbeitsaufträge

- minimiert die Antwortzeit des aperiodischen Arbeitsauftrags am Kopf oder die mittlere Antwortzeit aller Arbeitsaufträge in der AJQ
- lässt jeden neu ausgelösten sporadischen Arbeitsauftrag zu und plant diesen korrekt ein[20]
  - d.h., dass sein Abschluss zum vorgegebenen Termin sichergestellt ist

[20] Optimal ist das Verfahren (*online*) nur dann, wenn *alle* sporadischen Arbeitsaufträge planbar sind. Wird nur einer abgewiesen, gilt es als „nicht optimal“. [2, S. 192]

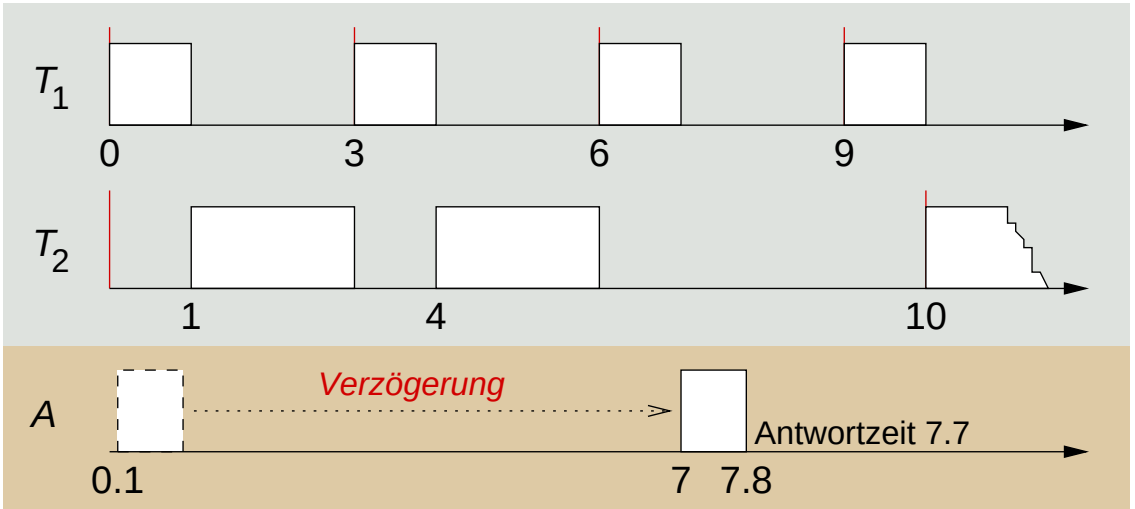
Hintergrundbetrieb

Korrekter Ablaufplan — auf Kosten des Antwortverhaltens

- aperiodische Arbeitsaufträge werden nur dann ausgeführt, wenn keine periodischen/sporadischen Arbeitsaufträge zur Ausführung anstehen

Beispiel:

- periodische Tasks
  - $T_1 = (3, 1)$
  - $T_2 = (10, 4)$
  - RM
- aperiodischer Job
  - $A \mapsto 0.8(0.1, \infty)$



- minimiert die (mittleren) Antwortzeiten aperiodischer Arbeitsaufträge nur suboptimal $\rightsquigarrow$ schlechtes Antwortverhalten

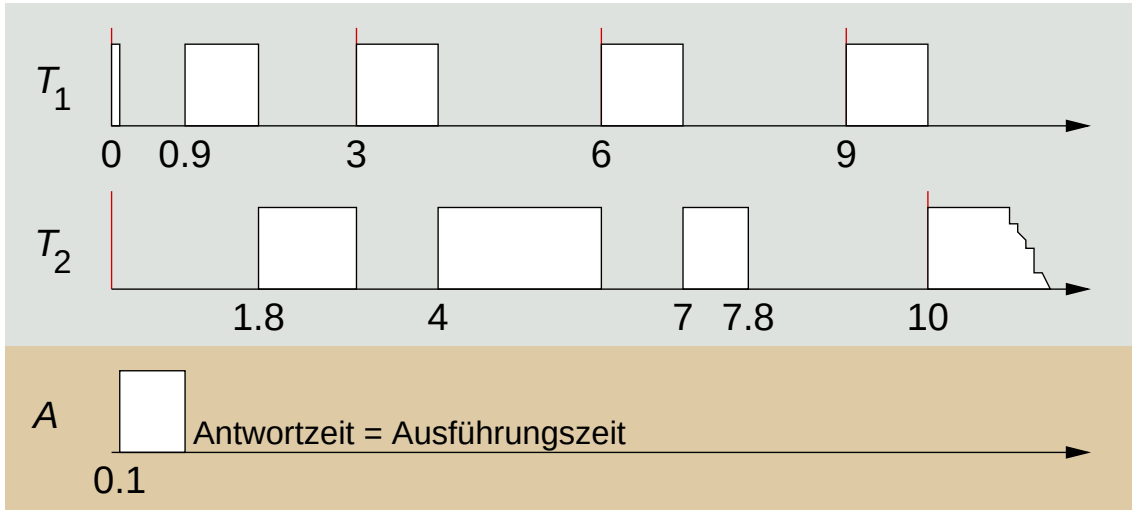
Unterbrecherbetrieb

Antwortzeitminimierung — auf Kosten eines „gut geordneten“ Ablaufplans

- ausgelöste aperiodische Arbeitsaufträge werden sofort ausgeführt, sie verdrängen die sich in Ausführung befindliche periodische Aufgabe

Beispiel:

- periodische Tasks
  - $T_1 = (3, 1)$
  - $T_2 = (10, 4)$
  - RM
- aperiodischer Job
  - $A \mapsto 0.8(0.1, \infty)$



- werden aperiodische Arbeitsaufträge immer sofort ausgeführt, erhöht sich das Risiko von Schwankungen im Ablauf periodischer Aufgaben

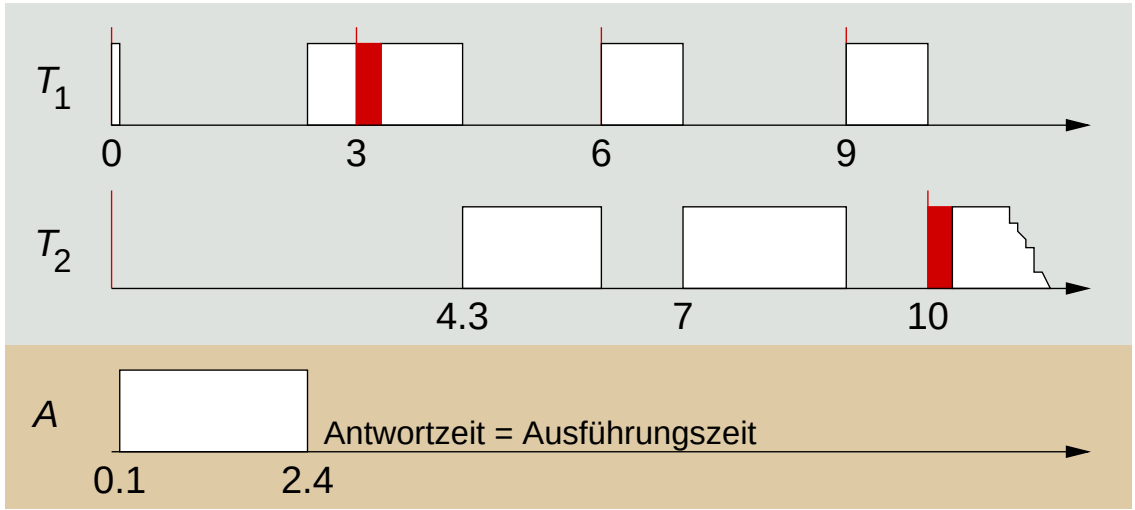
Unterbrecherbetrieb (Forts.)

Antwortzeitminimierung — auf Kosten eines korrekten Ablaufplans

- werden aperiodische Arbeitsaufträge bevorzugt ausgeführt, ist die Termineinhaltung periodischer Aufgaben nicht gesichert

Beispiel (vgl. 8- 6):

- aperiodischer Job
  - $A \mapsto 2.3(0.1, \infty)$
  - *run to completion*
- periodische Tasks
  - T_1 zu spät
  - T_2 zu spät



- ist die Schlupfzeit (S. 4- 9) der unterbrochenen periodischen Aufgabe abgelaufen, muss diese Aufgabe fortgesetzt werden

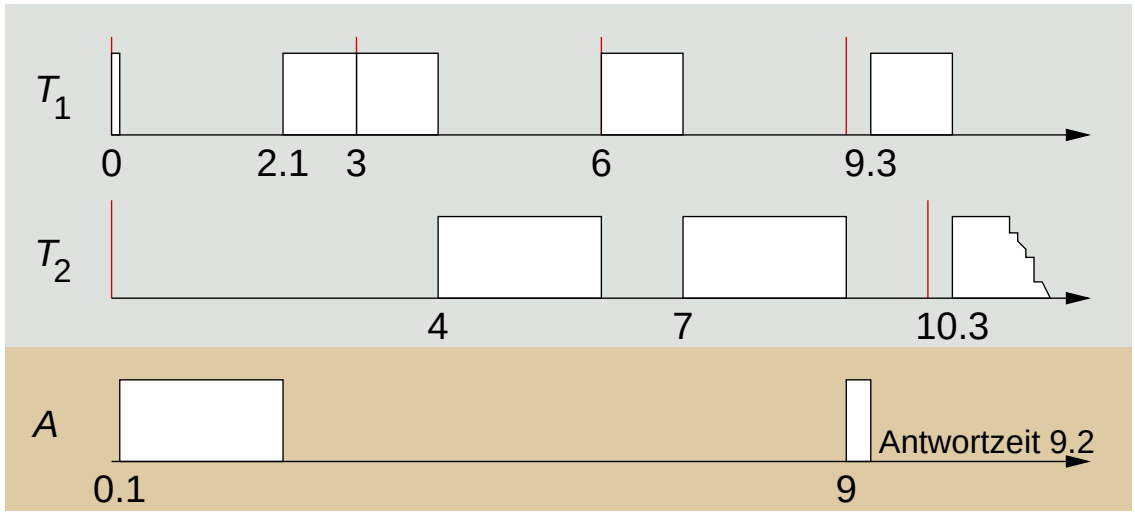
Unterbrecherbetrieb (Forts.)

Korrekter Ablaufplan — auf Kosten des Antwortverhaltens

- die Ausführung periodischer Aufgaben darf nur dann aufgeschoben werden, wenn der korrekte Ablaufplan weiterhin sicher ist

Beispiel (vgl. 8- 7):

- periodische Tasks
  - Schlupf stehlen
  - $slack(T_{1,2}) = 2$
- aperiodischer Job
  - Schlupf nutzen
  - unterbrechen



- *slack stealing* (S. 5- 25) minimiert die Antwortzeiten aperiodischer Arbeitsaufträge nur bedingt: $WCET(A) \leq min(slack(T_1), slack(T_2))$

Qual der Wahl...

Hintergrundbetrieb $\Longleftrightarrow$ Unterbrecherbetrieb

Hintergrundbetrieb

- \+ liefert immer korrekte Ablaufpläne
- \+ ist einfach zu implementieren
- – verlängert Antwortzeiten unnötigerweise

Unterbrecherbetrieb $\mapsto$ *slack stealing*

- \+ ist methodisch einfach in taktgesteuerten Systemen
  - – verkompliziert deren Implementierung jedoch beträchtlich
- – erweist sich als höchst komplex in ereignisgesteuerten Systemen

periodischer Zusteller (engl. *periodic server*) schafft Abhilfe

- eine (im korrekten Ablaufplan aufgestellte) periodische Aufgabe, die die bereitgestellten aperiodischen Arbeitsaufträge ausführt

Periodischer Zusteller (engl. periodic server)

Periodische Abarbeitung aperiodischer Arbeitsaufträge

Spezialisierung einer periodischen Aufgabe (S. 4- 12) von N aperiodischen Arbeitsaufträgen, $N > 0$ und variabel[21]

- definiert durch Periode p_s und Ausführungszeit e_s
  - das Verhältnis $u_s = e_s/p_s$ ist die Größe (engl. *size*) des Zustellers
- mit e_s als sogenanntes Ausführungsbudget (engl. *execution budget*)
  - das Budget wird um bis zu e_s Einheiten aufgefüllt (engl. *replenished*)
  - der Zeitpunkt dafür wird Auffüllzeit (engl. *replenishment time*) genannt
- innerhalb eines beliebigen Zeitintervals der Länge p_s niemals länger als e_s Zeiten (aperiodische Arbeitsaufträge) ausführend
  - Ausnahmen bestätigen diese Regel...
- verschiedenartig ausgelegt: abfragend, aufschiebbar, sporadisch

[21] N bestimmt die aktuelle Länge der AJQ.

Periodischer Zusteller (Forts.)
Arbeitsweise

Auslösung, Bereitstellung und Ausführung — der Zusteller...

- wird „zurückgestellt“ (engl. *backlogged*) wenn:
  1. die AJQ mindestens einen aperiodischen Arbeitsauftrag enthält
     - bei leerer AJQ ist der Zusteller untätig (engl. *idle*)
  2. der erste aperiodische Arbeitsauftrag in die leere AJQ kommt
- kommt in Frage (engl. *is eligible*) für die Ausführung wenn er:
  1. einen Auftragsüberhang (engl. *backlog*) aufweist *und*
  2. über ein Ausführungsbudget verfügt
- nimmt am Einplanungsverfahren periodischer Aufgaben teil
  - als „normale“ periodische Aufgabe mit $T_s = (p_s, e_s)$
- verbraucht (engl. *consumes*) sein Budget während der Ausführung
  - bis es auf Null abgesunken d.h. aufgebraucht (engl. *exhausted*) ist

Abfragender Zusteller (engl. *polling server*)

Verfall des Restbudgets zum Zeitpunkt des Untätigwerdens

Abfrager (engl. *poller*) $\mapsto T_P = (p_s, e_s)$

- mit Abfrageperiode p_s (engl. *polling period*)
  - zyklisch bereitgestellt im Abstand von (ganzzahlig vielfachen) p_s
- schrittweiser Abbau der AJQ innerhalb einer Abfrageperiode
  - Abarbeitung des Auftragsüberhangs vom Kopf der AJQ ausgehend
  - Unterbrechung des laufenden Arbeitsauftrags am Periodenende
- bei leerer AJQ verfällt das Budget unverzüglich
  - d.h., sobald der Abfrager feststellt, untätig sein zu müssen
  - auch, wenn dies bereits am Anfang der Abfrageperiode erkannt wird[22]
- Antwortzeiten aperiodischer Arbeitsaufträge schwanken ggf. stark
  - je nach Auslösezeitpunkt eines Auftrags bzw. Zustand des Abfragers

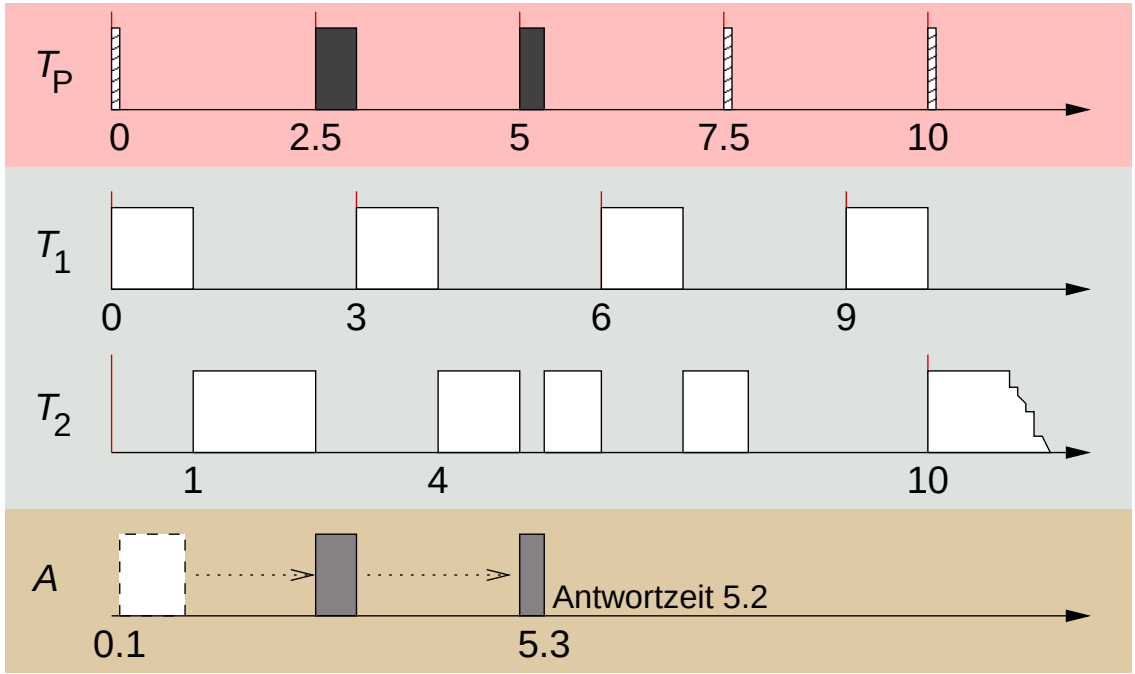
[22] Aperiodische Arbeitsaufträge, die eintreffen nachdem der Abfrager seine Untätigkeit festgestellt hat, kommen frühestens in der nächsten Abfrageperiode zum Zuge.

Abfragender Zusteller (Forts.)

Verbesserung des Antwortverhaltens

Beispiel (vgl. 8- 5):

- periodische Tasks
  - $T_P = (2.5, 0.5)$
  - $T_1 = (3, 1)$
  - $T_2 = (10, 4)$
  - RM
- aperiodischer Job
  - $A \mapsto 0.8(0.1, \infty)$



- T_P hat die kürzeste Periode und erhält daher höchste Priorität (RM)
- zu Beginn der Abfrageperioden t_0, $t_{7.5}$ und t_{10} ist die AJQ leer
- die Ausführung von A erfolgt in zwei Schritten:
  0.5 Zeiteinheiten (dem Budget von T_P) in Abfrageperiode $t_{2.5}$
  0.3 Zeiteinheiten in Abfrageperiode t_5, bis A beendet (und die AJQ leer) ist

Abfragender Zusteller (Forts.)

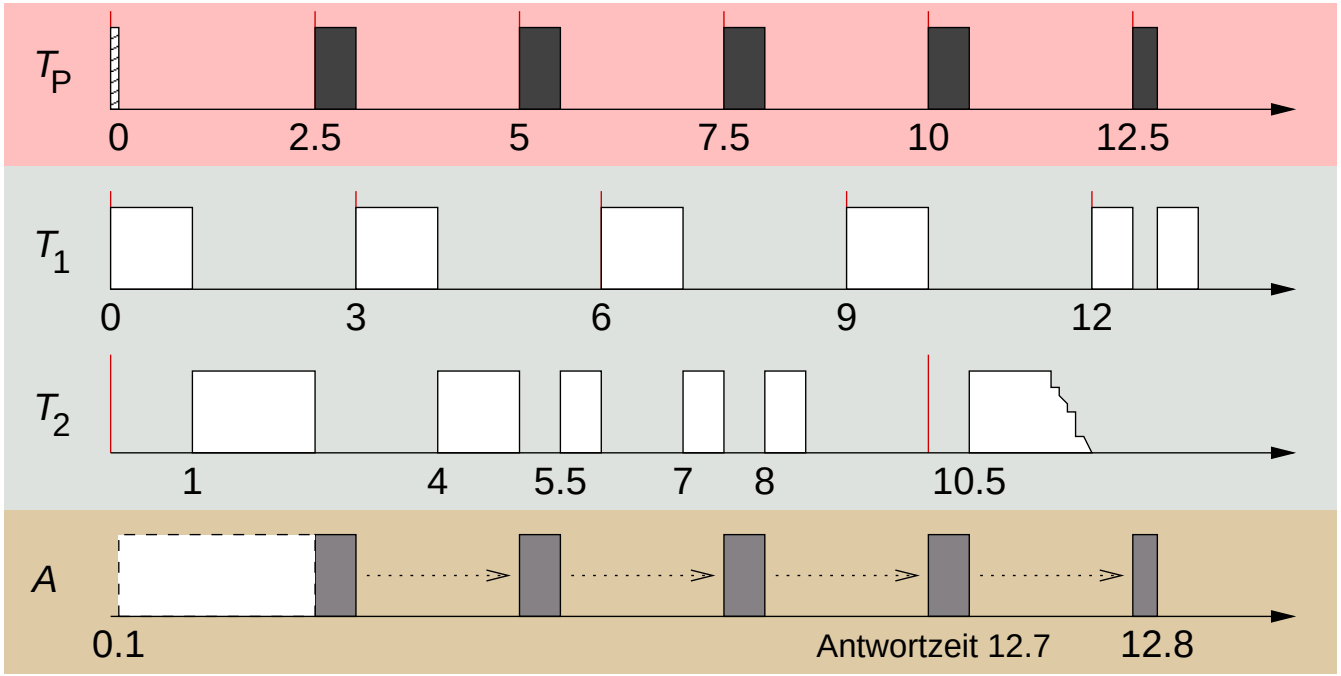
Verschlechterung des Antwortverhaltens

Beispiel (vgl. 8- 8):

T_P
T_1
T_2
} wie gehabt

RM

$A \mapsto 2.3(0.1, \infty)$



- die Ausführung von A benötigt (mindestens) fünf Abfrageperioden:
  4×0.5 Zeiteinheiten (dem Budget von T_P)
  - in den Abfrageperioden $t_{2.5}$, t_5, $t_{7.5}$ und t_{10}

  1×0.3 Zeiteinheiten (bis A beendet ist)
  - in Abfrageperiode $t_{12.5}$; die AJQ ist leer, T_P gibt auf...

Nachteil vom Abfragebetrieb

Schwankungen im Antwortverhalten des Systems

Verfall des noch nicht ganz ausgeschöpften Ausführungsbudgets eines untätigen Abfragers $\leadsto$ ggf. längere Antwortzeiten

- in der laufenden Abfrageperiode dann noch eintreffende aperiodische Arbeitsaufträge bleiben zunächst unberücksichtigt
  - sie sammeln sich in der AJQ an, der Abfrager wird zurückgestellt
  - so geschehen mit $A \mapsto 0.8(0.1, \infty)$, vgl. 8- 13
- die „zu spät“ eingetroffenen Aufträge werden frühestens in der nächsten Abfrageperiode (entsprechend Ablaufplan) behandelt

☞ das Restbudget eines Abfragers müsste bewahrt werden können. . .

Verbesserung des Abfragebetriebs

Restbudget untätig gewordener Zusteller in der Abfrageperiode nicht verfallen lassen

Zusteller, die ihr Ausführungsbudget (d.h., ihre Bandweite) innerhalb ihrer Abfrageperiode bewahren $\mapsto$ engl. *bandwidth-preserving server*

- definieren sich durch bestimmte Regeln zum...
  Verbrauch (engl. *consumption*)
  - Bedingungen, unter denen das Budget bewahrt/verbraucht wird

  Auffüllen (engl. *replenishment*)
  - Festlegungen, *wann* und *wie* das Budget aufgefüllt wird
- werden nach folgendem Schema vom System verarbeitet:
  - der Planer (Betriebssystem) führt Buch über den Budgetverbrauch
    - suspendiert den Zusteller, wenn das Budget verbraucht wurde
    - stellt den Zusteller bereit, wenn das Budget aufgefüllt wurde
  - der Zusteller setzt sich selbst aus, wenn er eine leere AJQ vorfindet
    - das Restbudget zum Zeitpunkt des Untätigwerdens bleibt ihm erhalten
    - sobald sich die AJQ wieder füllt, wird der Zusteller bereit gestellt

Aufschiebbarer Zusteller (engl. *deferrable server*)

Bewahrung des Restbudgets zum Zeitpunkt des Untätigwerdens

Deferrable Server $\mapsto T_D = (p_s, e_s)$

- periodisches Auffüllen von Budget e_s mit Periode p_s (vgl. 8- 13)
- bei leerer AJQ, Bewahrung des (Rest-) Budgets von T_D in p_s
- keine Akkumulation des Restbudgets von Periode zu Periode
  - am Ende der Abfrageperiode verfällt ein ggf. vorhandenes Restbudget

Verbrauchsregel Wann immer der Zusteller ausgeführt wird verbraucht sich das Ausführungsbudget des Zustellers mit einer Rate $1/\mathit{Zeiteinheit}$.

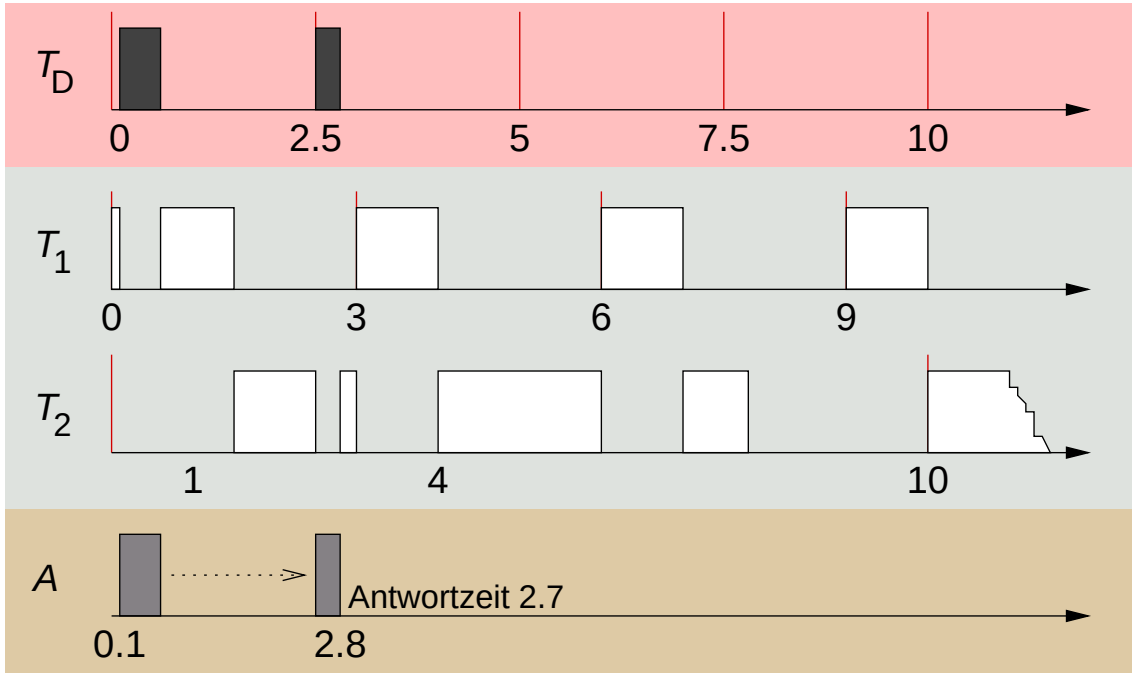
Auffüllregel Das Ausführungsbudget des Zustellers wird zu den Zeitpunkten kp_k auf e_s gesetzt, für $k = 0, 1, 2, \ldots$.

Aufschiebbarer Zusteller (Forts.)

Optimiertes Antwortverhalten

Beispiel (vgl. 8- 13):

- periodische Tasks
  - $T_D = (2.5, 0.5)$
  - $T_1 = (3, 1)$
  - $T_2 = (10, 4)$
  - RM
- aperiodischer Job
  - $A \mapsto 0.8(0.1, \infty)$



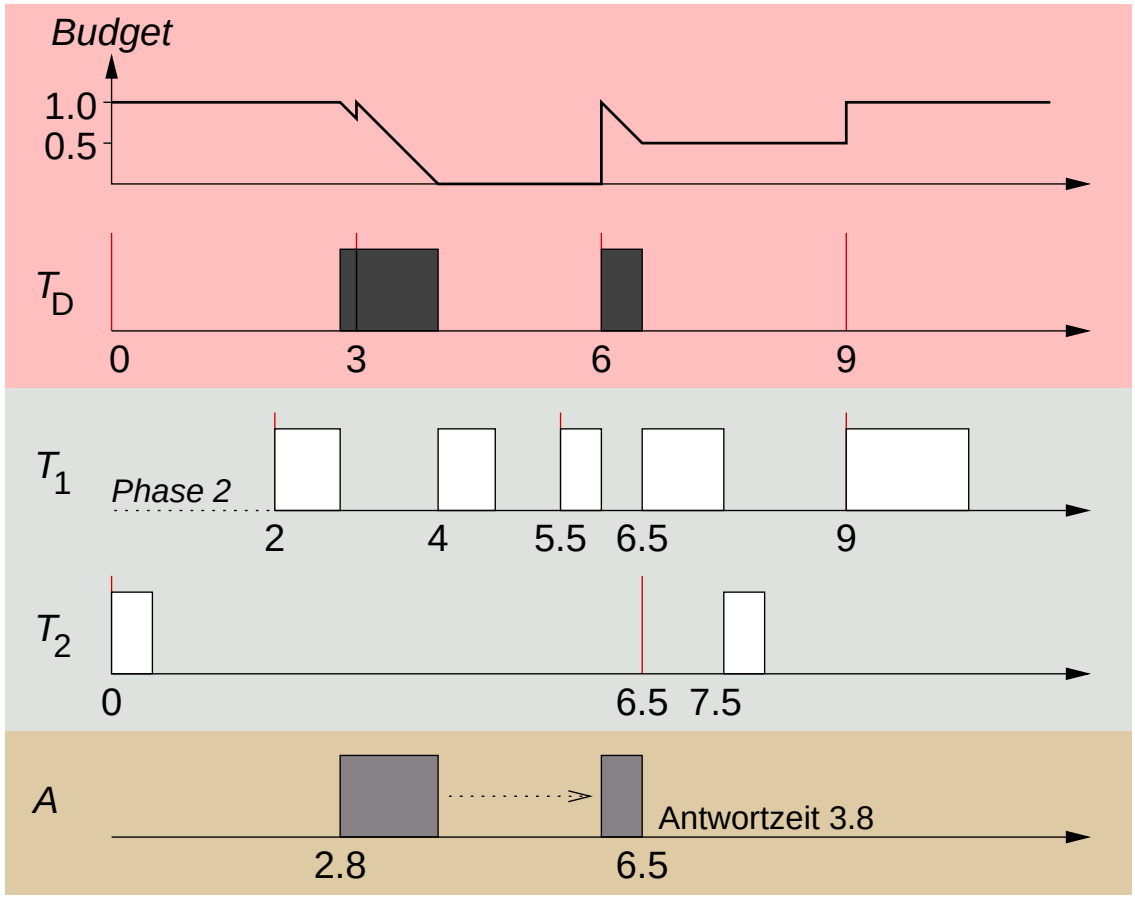
- die Ausführung von A erfolgt in zwei Schritten:
  - 0.5 Zeiteinheiten (dem Budget von T_D) in Abfrageperiode t_0
  - 0.3 Zeiteinheiten in Abfrageperiode $t_{2.5}$, dann setzt sich A selbst aus
- das Restbudget von 0.2 Zeiteinheiten bleibt T_D erhalten
  - es ist für nachrückende Aufträge in Abfrageperiode $t_{2.5}$ nutzbar
  - eine Übertragung auf Abfrageperiode t_5 ist nicht vorgesehen

Aufschiebbarer Zusteller (Forts.)

Budgetverbrauch und -auffüllung

Beispiel:

- periodische Tasks
  - $T_D = (3, 1)$
  - $T_1 = (2, 3.5, 1.5)$
  - $T_2 = (6.5, 0.5)$
  - RM
- aperiodischer Job
  - $A \mapsto 1.7(2.8, \infty)$



Verlauf:

t_0 T_D startet & wartet

$t_{2.8}$ A wird zugestellt

t_3 T_D kommt weiter in Frage

t_4 T_D wird vom Planer gestoppt

t_6 T_D kommt erneut in Frage

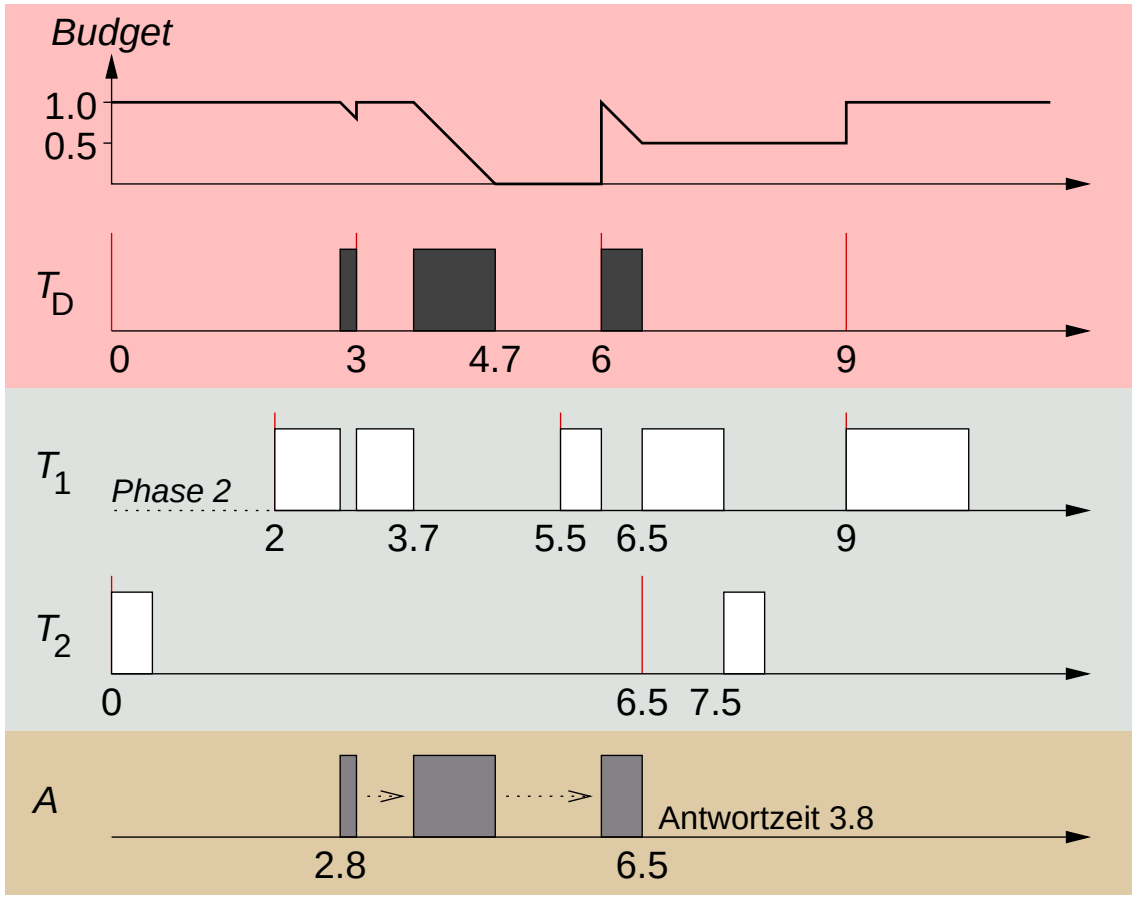
$t_{6.5}$ A ist beendet, T_D untätig

Aufschiebbarer Zusteller (Forts.)

Budgetverbrauch und -auffüllung — alternatives Einplanungsverfahren

Beispiel:

- periodische Tasks
  - $T_{D,1,2}$ vgl. 8- 19
  - EDF
  - $d_D = T_{replenishment}$
  - $d_{1,2} = p_{1,2}$
- aperiodischer Job
  - $A \mapsto 1.7(2.8, \infty)$



Verlauf:

t_0 T_D startet & wartet

$t_{2.8}$ A wird zugestellt

t_3 T_1 hat früheren Termin (5.5)

t_4 T_D wird weiter ausgeführt

t_6 $d_1 = d_D$, T_D wird bevorzugt

$t_{6.5}$ A ist beendet, T_D untätig

Aufschiebbarer Zusteller ∪ Hintergrundzusteller

Budgetverbrauch und -auffüllung — Antwortzeitverbesserung

Beispiel (vgl. 8- 20):

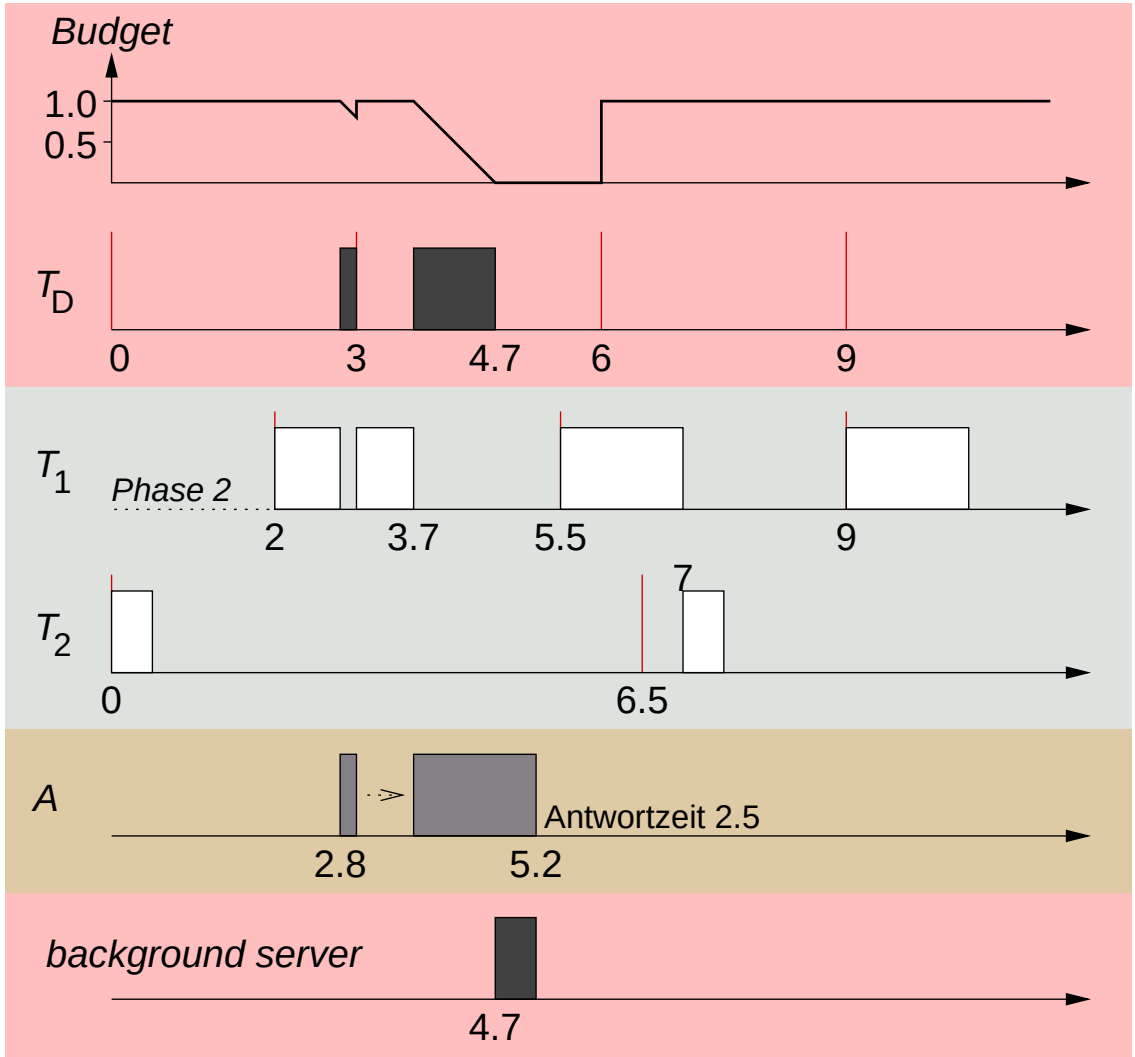
Background Server

- verarbeitet die AJQ
  - unterstützt T_D
  - Abfragervariante
- niedrigste Priorität

Verlauf:

$t_{4.7}$ keine periodische Task ist ausführbereit
- Hintergrundbetr.

$t_{5.2}$ A ist beendet
- T_D bleibt untätig



Aufschiebbarer Zusteller — Größenbeschränkung

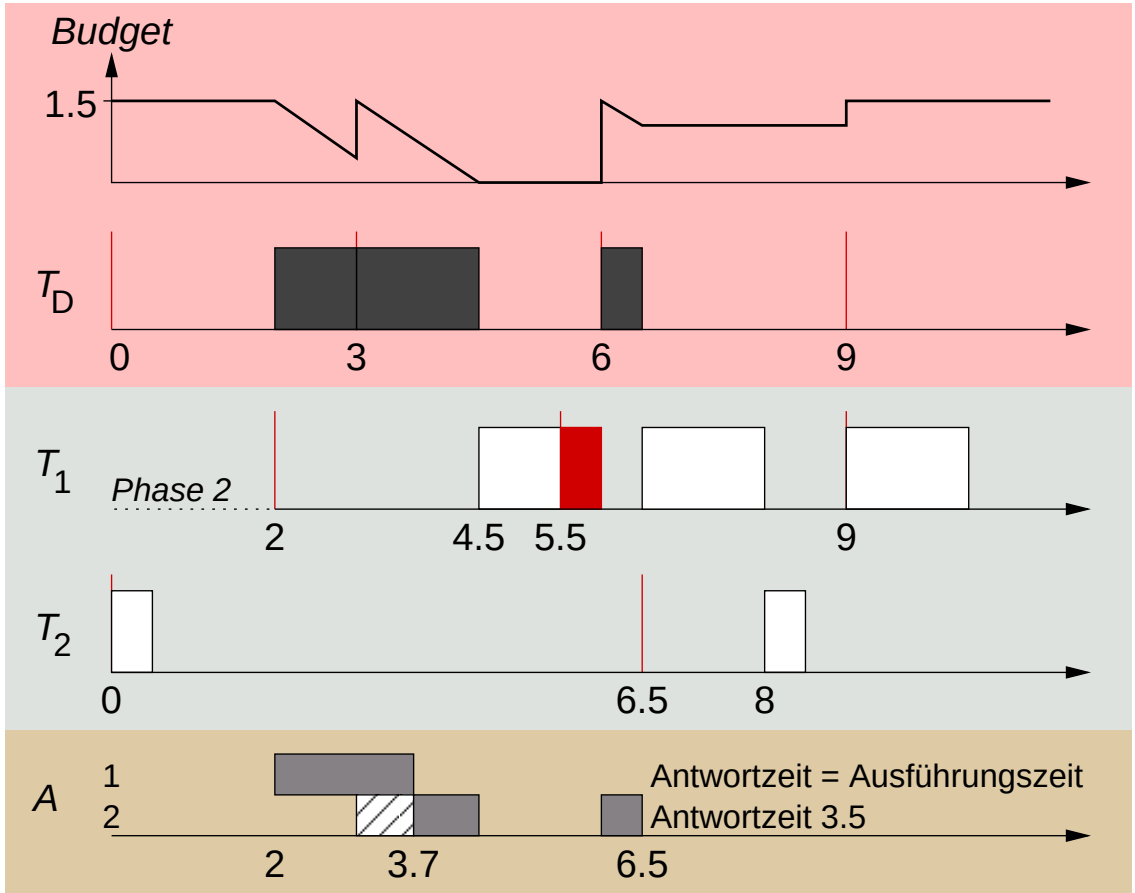
Einfluss auf die Planbarkeit periodischer Aufgaben

Verbesserung der Ansprechempfindlichkeit durch eine Vergrößerung des Budgets, anstatt Einsatz eines Hintergrundzusteller, ist problematisch:

Beispiel (vgl. 8- 19):

- $T_D = (3, 1.5)$
- $A \mapsto 3(2, \infty)$
- T_1 verpasst Termin

Das Budget ist unter Berücksichtigung aller möglichen Kombinationen von Auslösezeiten aller (periodischen) Tasks zu bestimmen.



Sporadischer Zusteller (engl. *sporadic server*)

Tasksysteme fester Priorität (engl. *fixed-priority systems*)

Sporadic Server $\mapsto T_S = (p_s, e_s)$

- beansprucht niemals mehr Prozessorzeit als die periodische Aufgabe $T = (p_s, e_s)$ in jedem Zeitintervall
- kann daher auch genau wie die periodische Aufgabe T behandelt werden, wenn auf Planbarkeit des Tasksystems geprüft wird
- ermöglicht die Planbarkeit eines Systems periodischer Aufgaben, das bei Verwendung eines aufschiebbaren Zustellers nicht planbar wäre
- kommt in verschiedenen Ausführungen vor, die sich im Wesentlichen in ihren Verbrauchs- und Auffüllregeln unterscheiden:

einfach	(engl. *simple*)	✓
kumulativ	(engl. *cumulative*) längere Budgetbewahrung	
SpSL	(Sprunt, Sha & Lehoczky) aggressiveres Auffüllen	
termingesteuert	(engl. *deadline-driven*) läuft mit höherer Priorität	

Sporadischer Zusteller (Forts.)
Definitionen

T Tasksystem fester Priorität
- von n unabhängigen, verdrängbaren periodischen Aufgaben

π_s beliebige Priorität des Zustellers T_S
- bei gleicher Priorität wird zu Gunsten von T_S entschieden

$\mathbf{T}_H$ Teilmenge von Tasks mit Prioritäten höher als π_s
- bleibt tätig[23] in jedem Tätigkeitsintervall von $\mathbf{T}_H$

Tätigkeitsintervall des Zustellers (engl. *server busy interval*)

beginnt mit Einspeisung eines Arbeitsauftrags in die leere AJQ

endet bei erneut leergelaufener AJQ

[23] **T** (oder $\mathbf{T}_H$) ist untätig, wenn kein Arbeitsauftrag in **T** (oder $\mathbf{T}_H$) ausführbereit ist; sonst ist **T** (oder $\mathbf{T}_H$) tätig.

Sporadischer Zusteller (Forts.)
Notationen

t_r späteste (aktuelle) Auffüllzeit

t_f erster Moment nach t_r, wenn die Ausführung von T_S beginnt

t_e späteste effektive Auffüllzeit

Zu jedem Zeitpunkt t ist:

t_{begin} der Anfang des frühesten Tätigkeitsintervalls der vor t liegenden spätesten zusammenhängenden Folge solcher Intervalle von $\mathbf{T}_H$

- zwei Tätigkeitsintervalle sind zusammenhängend, wenn das spätere direkt nach dem Ende des früheren startet

t_{end}

- das Ende des spätesten Tätigkeitsintervalls (der oben definierten Folge), wenn dieses Intervall vor t endet
- gleich ∞, wenn dieses Intervall nach t endet

Sporadischer Zusteller (Forts.)
Aktionen des Planers

1. t_r nimmt die aktuelle Zeit $t_{current}$ an, jedesmal wenn das Budget von T_S aufgefüllt wird
2. wenn T_S das erste Mal nach Auffüllung seines Budgets mit der Ausführung startet, wird:
   - 2.1 t_e auf Basis der Historie des Tasksystems bestimmt
   - 2.2 die nächste Auffüllzeit auf $t_e + p_s$ gesetzt

Mit anderen Worten: die nächste Auffüllzeit ist p_s Zeiteinheiten entfernt von t_e,

- als wenn das Budget zuletzt zum Zeitpunkt t_e aufgefüllt worden wäre
- daher der Begriff „effektive Auffüllzeit“

Einfacher sporadischer Zusteller

Implementierungsmöglichkeit und Verbrauchsregel

Planer

- überwacht die Tätigkeitsintervalle von $\mathbf{T}_H$
- pflegt Informationen über t_{begin} und t_{end}

Verbrauchsregel zu jedem Zeitpunkt $t > t_r$ (bis das Budget erschöpft ist) verbraucht sich das Ausführungsbudget des Zustellers mit einer Rate $1/\textit{Zeiteinheit}$, wenn gilt:

V1 der Zusteller wird ausgeführt

V2 der Zusteller wurde seit t_r ausgeführt und $t_{end} < t$

Ansonsten behält der Zusteller sein Budget.

Einfacher sporadischer Zusteller (Forts.)

Auffüllregel

Auffüllregel

A1 initial (d.h., wenn die Ausführung von **T** beginnt) und jedesmal, wenn das Budget aufgefüllt wird:

- $Budget = e_s$ und $t_r = t_{current}$

A2 zum Zeitpunkt t_f,

$$t_e = \begin{cases} max(t_r, t_{begin}) & \text{wenn } t_{end} = t_f \\ t_f & \text{wenn } t_{end} < t_f \end{cases}$$

- $t_e + p_s$ ist nächste Auffüllzeit

A3 nächste Auffüllung erfolgt bei $t_e + p_s$, es sei denn,

(a) $t_e + p_s < t_f$: Auffüllung, wenn Budget erschöpft ist

(b) **T** wird untätig vor $t_e + p_s$ und wieder tätig bei t_b: Auffüllung zum Zeitpunkt $min(t_e + p_s, t_b)$

Einfacher sporadischer Zusteller (Forts.)

Interpretation von Verbrauchs- und Auffüllregel

V1 ist selbsterklärend

V2
- T_S verbraucht sein Budget bis zu einem beliebigen Zeitpunkt t, wenn er seit t_r tätig war
- aber zum Zeitpunt t wird T_S suspendiert und $\mathbf{T}_H$ untätig

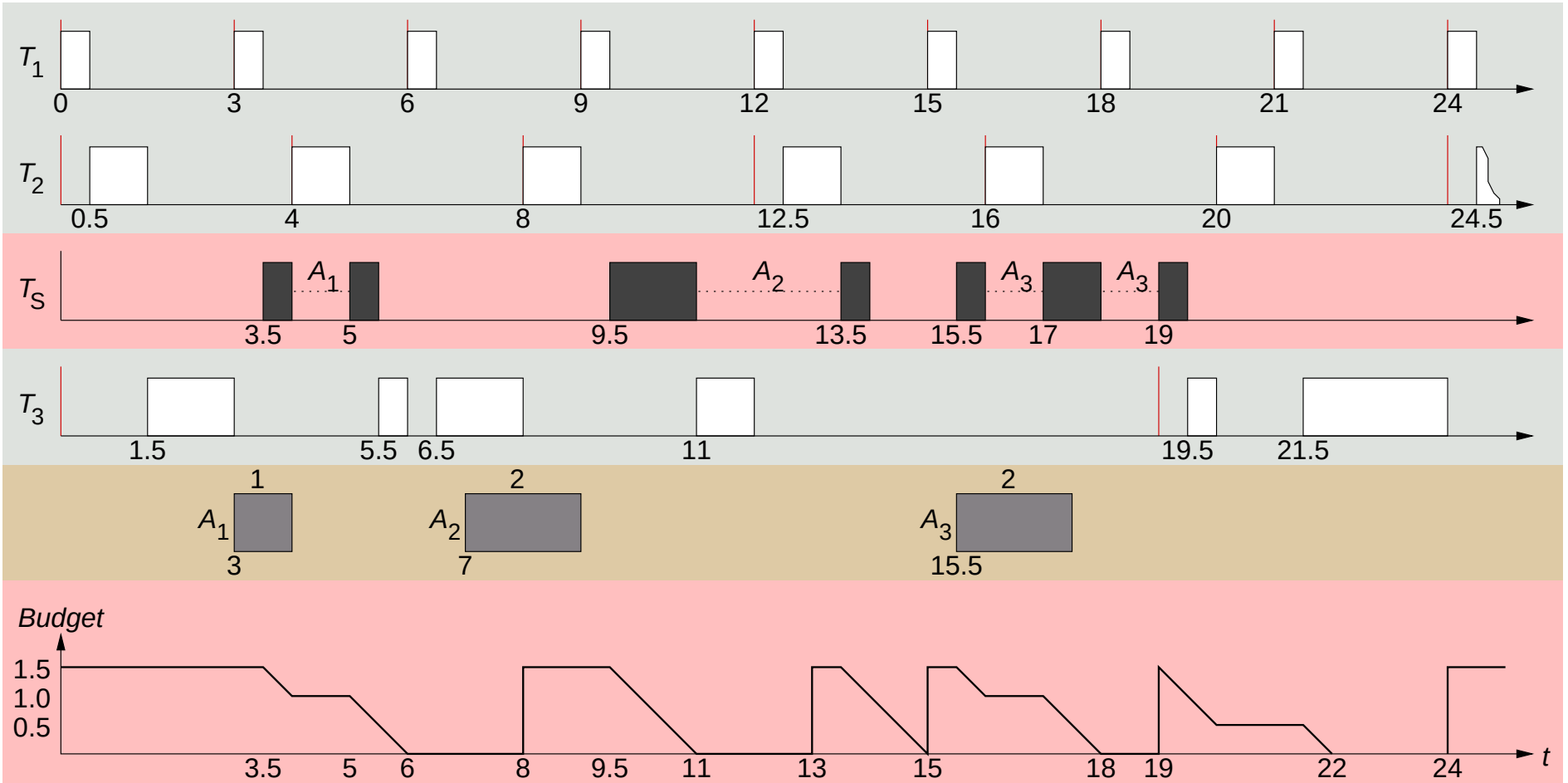
A1 ist selbsterklärend

A2
- die nächste Budgetauffüllung erfolgt p_s Zeiteinheiten nach t_r, wenn $\mathbf{T}_H$ im Intervall (t_r, t_f) tätig war
- sonst liegt t_e später; t_e ist spätester Moment in (t_r, t_f), zu dem:
  - eine gleich- oder niederpriore Task ausgeführt wird *oder*
  - $\mathbf{T}$ untätig ist

A3 siehe Beispiel 8- 30

Einfacher sporadischer Zusteller — Beispiel

$T_1 = (3, 0.5)$, $T_2 = (4, 1)$, $T_3 = (19, 4.5)$ und $T_S = (5, 1.5)$; RM



Einfacher sporadischer Zusteller — Beispiel (Forts.)

Budgetverbrauch und -auffüllung

$t_{3.5}$ T_S startet: $t_r = 0$, $t_{begin} = 3$, $t_{end} = 3.5$

- wegen A2 gilt: $t_e = max(0, 3) = 3$, nächste Auffüllung bei $t = 8$

t_4 T_2 verdrängt T_S, der sein Budget bewahrt

t_5 T_S wird fortgesetzt und verbraucht sein Budget komplett

- da er läuft (V1) und, wenn er wieder suspendiert wird, T_1 und T_2 untätig sind bzw. $t_{end} = 5 < t_{current}$ (V2)

t_8 Budgetauffüllung (A3), T_S ausführbereit

$t_{9.5}$ T_S startet zum ersten Mal seit $t = 8$

- $t_e = 8$, nächste Auffüllung bei $t = t_e + p_s = 13$

t_{14} **T** untätig; nächste Auffüllung bei $t = 15$

- wenn das nächste Tätigkeitsintervall von **T** beginnt
- $t_e = 18$, $t_b = 15$: $min(t_e + p_s, t_b) = 15$ (A3(b))

t_{19} Budgetauffüllung (A3(b))

Resümee

Einführung

- Übernahmeprüfung, Anwortzeitminimierung
- Warteschlangen, AJQ; korrekter/optimaler Ablaufplan

Problemfälle

- Hintergrundbetrieb, Unterbrecherbetrieb

periodische Zusteller

- Arbeitsweise: Auslösung, Bereitstellung, Ausführung

Bandweite verlierende Zusteller $\mapsto$ Abfrager

- Problem: Verfall des Restbudgets bei Untätigkeit

Bandweite bewahrende Zusteller $\rightsquigarrow$ Verbrauchs-/Auffüllregeln

- aufschiebbare: ohne/mit Hintergrundzusteller, Planbarkeit
- sporadische: *einfach*; kumulativ, SpSL, termingesteuert